Generalized Majorization-Minimization

Sobhan Naderi Parizi sobhan@alumni.brown.edu
School of Engineering, Brown University
Providence, RI 02912, USA

Kun He hekun@bu.edu
Dept. of Computer Science, Boston University
Boston, MA 02215, USA

Stan Sclaroff sclaroff@bu.edu
Dept. of Computer Science, Boston University
Boston, MA 02215, USA

Pedro Felzenszwalb pff@brown.edu
School of Engineering, Brown University
Providence, RI 02912, USA

Abstract

Non-convex optimization is ubiquitous in machine learning. The Majorization-Minimization (MM) procedure systematically optimizes non-convex functions through an iterative construction and optimization of upper bounds on the objective function. The bound at each iteration is required to *touch* the objective function at the optimizer of the previous bound. We show that this touching constraint is unnecessary and overly restrictive. We generalize MM by relaxing this constraint, and propose a new optimization framework, named Generalized Majorization-Minimization (G-MM) that is more flexible compared to MM. For instance, it can incorporate application-specific biases into the optimization procedure without changing the objective function. We derive G-MM algorithms for several latent variable models and show empirically that they consistently outperform their MM counterparts in optimizing non-convex objectives. In particular, G-MM algorithms appear to be less sensitive to initialization.

Keywords: majorization-minimization, non-convex optimization, latent variable models, expectation maximization

1. Introduction

Non-convex optimization is ubiquitous in machine learning. For example, data clustering (MacQueen, 1967; Arthur and Vassilvitskii, 2007), training classifiers with latent variables (Yu and Joachims, 2009; Felzenszwalb et al., 2010; Pirsiavash and Ramanan, 2014; Azizpour et al., 2015),and training visual object detectors from weakly labeled data (Song et al., 2014; Rastegari et al., 2015; Ries et al., 2015) all lead to non-convex optimization problems.

Majorization-Minimization (MM) (Hunter et al., 2000) is an optimization framework for designing well-behaved optimization algorithms for non-convex functions. MM algorithms work by iteratively optimizing a sequence of easy-to-optimize surrogate functions that bound the objective. Two of the most successful instances of MM algorithms are Expectation-Maximization (EM) (Dempster et al., 1977) and the Concave-Convex Proce-

dure (CCCP) (Yuille and Rangarajan, 2003). However, both have a number of drawbacks in practice, such as sensitivity to initialization and lack of uncertainty modeling for latent variables. This has been noted in works such as (Neal and Hinton, 1998; Felzenszwalb et al., 2010; Parizi et al., 2012; Kumar et al., 2012; Ping et al., 2014).

We propose a new procedure, *Generalized Majorization-Minimization (G-MM)*, for optimizing non-convex objective functions. Our approach is inspired by MM, but we generalize the bound construction process. Specifically, we relax the strict bound construction method in MM to allow for a *set* of valid bounds to be used, while still maintaining algorithmic convergence. This relaxation gives us more freedom in bound selection and can be used to design better optimization algorithms.

In training latent variable models and in clustering problems, MM algorithms such as CCCP and k-means are known to be sensitive to the initial values of the latent variables or cluster memberships. We refer to this problem as *stickiness* of the algorithm to the initial latent values. Our experimental results show that G-MM leads to methods that tend to be less sticky to initialization. We demonstrate the benefit of using G-MM on multiple problems, including k-means clustering and applications of Latent Structural SVMs to image classification with latent variables.

1.1 Related Work

Perhaps the most famous iterative algorithm for non-convex optimization in machine learning and statistics is the EM algorithm (Dempster et al., 1977). EM is best understood in the context of maximum likelihood estimation in the presence of missing data, or *latent variables*. EM is a bound optimization algorithm: in each E-step, a lower bound on the likelihood is constructed, and the M-step maximizes this bound.

Countless efforts have been made to extend the EM algorithm since its introduction. In (Neal and Hinton, 1998) it is shown that, while both steps in EM involve optimizing some functions, it is not necessary to fully optimize the functions in each step; in fact, each step only needs to "make progress". This relaxation can potentially avoid sharp local minima and even speed up convergence.

In another attempt, Hunter et al. (2000) proposed the Majorization-Minimization (MM) framework. MM generalizes methods like EM by "transferring" the optimization to a sequence of surrogate functions (bounds) on the original objective function. The Concave-Convex Procedure (CCCP) (Yuille and Rangarajan, 2003) is another widely-used instance of MM, where the surrogate function is obtained by *linearizing* the concave part of the objective function. Many successful machine learning algorithms employ CCCP, *e.g.* the Latent SVM (Felzenszwalb et al., 2010). Other instances of MM algorithms include iterative scaling (Pietra et al., 1997), and non-negative matrix factorization (Lee and Seung, 1999). Another related line of research concerns the Difference-of-Convex (DC) programming (Tao, 1997), which can be shown to reduce to CCCP under certain conditions. Convergence properties of such general "bound optimization" algorithms have been discussed by Salakhutdinov et al. (2002).

Despite widespread success, MM (and CCCP in particular) has a number of drawbacks, some of which have motivated our work. In practice, CCCP often exhibits *stickiness* to initialization, which necessitates expensive initialization or multiple trials (Parizi et al., 2012;

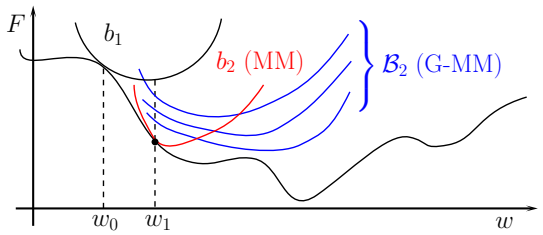


Figure 1: Optimization of a function F using MM (red) and G-MM (blue). In MM the bound b_2 has to touch F at w_1. In G-MM we only require that b_2 be below b_1 at w_1, leading to several choices $\mathcal{B}_2$.

Algorithm 1: G-MM optimization

Input: w_0, η, ϵ
$v_0 := F(w_0)$
for $t := 1, 2, \ldots$ **do**
  select $b_t \in \mathcal{B}_t = \mathcal{B}(w_{t-1}, v_{t-1})$ as in (4)
  $w_t := \operatorname{argmin}_w b_t(w)$
  $d_t := b_t(w_t) - F(w_t)$
  $v_t := b_t(w_t) - \eta d_t$
  if $d_t < \epsilon$ **break**
end for
Output: w_t

Song et al., 2014; Cinbis et al., 2016). In optimizing latent variable models, CCCP lacks the ability to incorporate application-specific information without making modifications to the objective function, such as prior knowledge or side information (Xing et al., 2002; Yu, 2012), latent variable uncertainty (Kumar et al., 2012; Ping et al., 2014), and posterior regularization (Ganchev et al., 2010). Our framework can deal with these drawbacks. Our key observation is that we can relax the constraint enforced by MM that requires the bounds to touch the objective function, and this relaxation gives us the ability to better avoid sensitivity to initialization, and to incorporate side information.

A closely related work to ours is the pseudo-bound optimization framework by Tang et al. (2014), which generalizes CCCP by using "pseudo-bounds" that may intersect the objective function. In contrast, our framework still uses valid bounds, but only relaxes the touching requirement. Also, the pseudo-bound optimization framework is not as general as ours in that it is designed specifically for optimizing binary energies in MRFs, and it restricts the form of surrogate functions to parametric max-flow.

The generalized variants of EM proposed and analyzed by Neal and Hinton (1998) and Gunawardana and Byrne (2005) are related to our work when we restrict our attention to probabilistic models and the EM algorithm. EM can be viewed as a bound optimization procedure where the likelihood function involves both the model parameters θ and a distribution q over the latent variables, denoted by $F(\theta, q)$. Choosing q to be the posterior leads to a lower bound on F that is tight at the current estimate of θ. Generalized versions of EM, such as those given by Neal and Hinton (1998), use distributions other than the posterior in an alternating optimization of F. This fits into our framework, as we use the exact same objective function, ans only change the bound construction step (which amounts picking the distribution q in EM). We propose both *stochastic* and *deterministic* strategies for bound construction, and demonstrate that they lead to higher quality solutions and less sensitivity to initialization than other EM-like methods.

2. Proposed Optimization Framework

We consider minimization of functions that are bounded from below. The extension to maximization is trivial. Let $F(w) : \mathbb{R}^d \to \mathbb{R}$ be a lower-bounded function that we wish

to minimize. We propose an iterative procedure that generates a sequence of solutions $w_0, w_1, \ldots$ until it converges. The solution at iteration $t \geq 1$ is obtained by minimizing an upper bound $b_t(w)$ to the objective function *i.e.* $w_t = \operatorname{argmin}_w b_t(w)$. The bound at iteration t is chosen from a set of "valid" bounds $\mathcal{B}_t$ (see Figure 1). In practice, we assume that the members of $\mathcal{B}_t$ come from a family $\mathcal{F}$ of functions that upper bound F and can be optimized efficiently, such as quadratic functions, or quadratic functions with linear constraints. Algorithm 1 gives the outline of the approach.

This general scheme is used in both MM and G-MM. However, as we shall see in the rest of this section, MM and G-MM have key differences in the way they measure progress and the way they construct new bounds.

2.1 Progress Measure

MM measures progress with respect to the objective values. To guarantee progress over time MM requires that the bound at iteration t must touch the objective function at the previous solution, leading to the following constraint:

$$\textbf{MM constraint:} \qquad b_t(w_{t-1}) = F(w_{t-1}). \tag{1}$$

This touching constraint, together with the fact that w_t minimizes b_t leads to $F(w_t) \leq F(w_{t-1})$. That is, the value of the objective function is non-increasing over time.

The touching requirement is restrictive and, in practice, can make it hard to avoid local minima. In particular, this can make MM algorithms such as CCCP sensitive to initialization (Parizi et al., 2012; Cinbis et al., 2016; Song et al., 2014). The touching constraint also eliminates the possibility of using bounds that do not touch the objective function but may have other desirable properties.

In G-MM, we measure progress with respect to the bound values. It allows us to relax the touching constraint of MM, stated in (1), and require instead that,

$$\textbf{G-MM constraints:} \quad \begin{cases} b_1(w_0) = F(w_0) \\ b_t(w_{t-1}) \leq b_{t-1}(w_{t-1}). \end{cases} \tag{2}$$

Note that the G-MM constraints are weaker than MM. In fact, (1) implies (2) because b_{t-1} is an upper bound on F.

The weaker constraints used in G-MM do not imply $F(w_t) \leq F(w_{t-1})$, but, are sufficient to ensure that $\forall t, F(w_t) \leq F(w_0)$. This follows from the fact that b_t is an upper bound of F, w_t is a minimizer of $b_t(w)$, and (2):

$$F(w_t) \leq b_t(w_t) \leq b_t(w_{t-1}) \leq \cdots \leq b_1(w_1) \leq b_1(w_0) = F(w_0). \tag{3}$$

2.2 Bound Construction

This section describes step 3 of Algorithm 1. To construct new bounds, G-MM considers a "valid" subset of the upper bounds in $\mathcal{F}$ that satisfy (2). We denote the set of valid bounds at iteration t by $\mathcal{B}_t$. We consider two scenarios for constructing bounds in G-MM. The first scenario is when we have a bias function $g : \mathcal{B}_t \times \mathbb{R}^d \to \mathbb{R}$ over the set of valid bounds. The function g takes in a bound $b \in \mathcal{B}_t$ and a current solution $w \in \mathbb{R}^d$ and returns a scalar

indicating the goodness of the bound. In this scenario we select the bound with the largest bias value *i.e.* $b_t = \text{argmax}_{b \in \mathcal{B}_t} g(b, w_{t-1})$.

In the second scenario we propose to choose one of the valid bounds from $\mathcal{B}_t$ at random. Thus, we have both a deterministic (the 1^{st} scenario) and a stochastic (the 2^{nd} scenario) bound construction mechanism. MM algorithms, such as EM and CCCP, fall into the first scenario.

In general MM algorithms are instances of G-MM that use a specific bias function $g(b, w) = -b(w)$. The bound that touches F at w maximizes this bias function. It also maximizes the amount of progress with respect to the previous bound at w. By choosing bounds that maximize progress, MM algorithms tend to rapidly converge to a nearby local minimum. For instance, at iteration t, the CCCP bound for latent SVMs is obtained by fixing latent variables in the concave part of the objective function to the best values according to the previous solution w_{t-1}, making the new solution $w_t = \text{argmin}_w b_t(w)$ attracted to w_{t-1}. Similarly, in the E-step, EM sets the posterior distribution of the latent variables conditioned on the data according to the model from the previous iteration. Thus, in the next maximization step, the model is updated to "match" the fixed posterior distributions. This explains one reason why MM algorithms are observed to be sticky to initialization.

G-MM offers a more flexible bound construction scheme than MM. In Section 5 we show empirically that picking bounds uniformly at random from the set of valid bounds is less sensitive to initialization and leads to better results compared to CCCP and EM. We also show that using good bias functions can further improve performance of the learned models.

To guarantee convergence, we restrict the set of valid bounds $\mathcal{B}_t$ to those that are below a threshold v_{t-1} at the previous solution w_{t-1}:

$$\begin{aligned} \mathcal{B}_t &= \mathcal{B}(w_{t-1}, v_{t-1}) \\ \mathcal{B}(w, v) &= \{b \in \mathcal{F} \mid b(w) \leq v\}. \end{aligned} \tag{4}$$

Initially, we set $v_0 = F(w_0)$ to ensure that the first bound *touches* F. In the next iterations, we set v_t using a *progress coefficient* $\eta \in (0, 1]$ that ensures making, at least, ηd_t progress where $d_t = b_t(w_t) - F(w_t)$ is the gap between the bound and the true objective value at w_t.

Note that when $\eta = 1$ all valid bounds touch F at w_{t-1} (see (4)), corresponding to the MM requirement. Small η values allow for gradual exploratory progress in each step while large η values greedily select bounds that guarantee immediate progress.

3. Convergence of G-MM

We showed in (3) that, in any iteration t, the G-MM solution is no worse than the initial solution in terms of the objective value, *i.e.* $F(w_t) \leq F(w_0)$. In this section we also show that the gap between the G-MM bounds and the true objective converges to zero at the minimizers of the bounds (Theorem 1). Moreover, we show, under mild conditions, G-MM converges to a solution (Theorem 2) that is a local extremum or a saddle point of F (Theorem 3). Other convergence properties of G-MM depend on the structure of the objective function F, the family of the bounds $\mathcal{F}$, and the details of the bound selection strategy.

Theorem 1. *In the limit as $t \to \infty$, G-MM bounds and F* ***touch*** *at the minimizer of the bounds,* i.e. $\lim_{t\to\infty} d_t = 0$*, where $d_t = b_t(w_t) - F(w_t)$.*

Proof. We have $b_t(w_t) \leq b_t(w_{t-1}) \leq v_{t-1}$, where v_t is defined as in line 6 of Algorithm 1. The first inequality holds because w_t minimizes b_t and the second inequality follows from (4). Summing over t gives

$$\sum_{t=1}^{T} b_t(w_t) \leq \sum_{t=1}^{T} v_{t-1} = v_0 + \sum_{t=1}^{T-1} b_t(w_t) - \eta d_t \tag{5}$$

$$\Rightarrow \ \eta \sum_{t=1}^{T} d_t \leq v_0 - b_T(w_T) = F(w_0) - b_T(w_T). \tag{6}$$

Let $w^* \in \operatorname{argmin}_w F(w)$ be an unknown *global* minimum of F. Using $F(w^*) \leq F(w_T) \leq b_T(w_T)$ and (6), we have

$$\eta \sum_{t=1}^{T} d_t \leq F(w_0) - F(w^*) \ \Rightarrow \ \lim_{t\to\infty} d_t = 0. \tag{7}$$

If a global minimizer of F does not exist, we can use the assumption that F is bounded from below and replace $F(w^*)$ in (7) with the value of the lower bound. □

Theorem 2. *If $\mathcal{F}$ is a family of m-strongly convex functions where $m > 0$, then $\exists w^\dagger$* s.t. $\lim_{t\to\infty} w_t = w^\dagger$*;* i.e. *G-MM converges to a solution.*

Proof. Let f be an m-strongly convex function, and let x, y be two points in the domain of f, also let $g \in \partial f$ be a subgradient of f. We have

$$f(y) \geq f(x) + g^T(y - x) + \frac{m}{2}||x - y||^2. \tag{8}$$

After substituting $f = b_t$, $x = w_t$, and $y = w_{t-1}$ in (8), and noting that the zero vector is a subgradient of b_t at w_t, *i.e.* $0 \in \partial b_t(w_t)$, we get:

$$\frac{m}{2}\left(b_t(w_{t-1}) - b_t(w_t)\right) \geq ||w_t - w_{t-1}||^2. \tag{9}$$

We can replace $b_t(w_{t-1})$ by v_{t-1} in (9) due to (4). By adding the inequalities for T iterations, similar to what we did in the proof of Theorem 1, we get:

$$\frac{m}{2}\left(F(w_0) - F(w^*)\right) \geq \sum_{t=1}^{T} ||w_t - w_{t-1}||^2 \tag{10}$$

$$\Rightarrow \exists w^\dagger \ s.t. \ \lim_{t\to\infty} w_t = w^\dagger. \tag{11}$$

Similar to the proof of Theorem 1, if a global minimum does not exist, we can replace $F(w^*)$ with the value of the lower bound of F in (10). □

Theorem 3. *Let F be continuously differentiable, and $\mathcal{F}$ be a family of strongly convex and smooth functions. Further, if $\exists \infty > M > m > 0, \forall b \in \mathcal{F}, MI \succeq \nabla^2 b(w) \succeq mI$, where I is the identity matrix, then $\nabla F(w^\dagger) = 0$; that is, G-MM converges to a local extremum or saddle point of F.*

We postpone the proof of Theorem 3 to the appendix.

4. Derived Optimization Algorithms

For simplicity and ease of exposition, we primarily focus on demonstrating G-MM on *latent variable models* where bound construction naturally corresponds to imputing latent variables in the model. In this section we derive G-MM algorithms for two models widely used in machine learning, namely, k-means clustering and Latent Structural SVM, both belonging to the category of latent variable models. It is worth mentioning that G-MM is fully capable of handling more general non-convex problems.

4.1 k-means Clustering

Let $\{x_1, \ldots, x_n\}$ denote a set of sample points and $w = (\mu_1, \ldots, \mu_k)$ denote a set of cluster centers. We use $z_i \in \{1, \ldots, k\}$ to denote the index of the cluster assigned to the i-th sample point. The objective function in k-means clustering is defined as follows,

$$F(w) = \sum_{i=1}^{n} \min_{z_i=1}^{k} ||x_i - \mu_{z_i}||^2, \quad w = (\mu_1, \ldots, \mu_k). \tag{12}$$

Bound construction: We obtain a convex upper bound on F by fixing the latent variables $(z_1, \ldots, z_n)$ to certain values instead of minimizing over these variables. Bounds constructed this way are quadratic convex functions of $w = (\mu_1, \ldots, \mu_k)$,

$$\mathcal{F} = \left\{ \sum_{i=1}^{n} ||x_i - \mu_{z_i}||^2 \;\middle|\; \forall i,\ z_i \in \{1, \ldots, k\}^n \right\}. \tag{13}$$

The k-means algorithm is an instance of MM methods. The algorithm repeatedly assigns each example to its nearest center to construct a bound, and then updates the centers by optimizing the bound. We can set $g(b, w) = -b(w)$ in G-MM to obtain the k-means algorithm. We can also define g differently to obtain a G-MM algorithm that exhibits other desired properties. For instance, a common issue in clustering is cluster starvation. One can design a bias function that encourages balanced clusters by selecting g appropriately.

We can select a random bound from $\mathcal{B}_t$ by sampling a latent configuration $z = (z_1, \ldots, z_n)$ uniformly from the set of configurations leading to valid bounds. Specifically, we start from a valid initial configuration (*e.g.* k-means solution) and do a random walk on a graph whose nodes are latent configurations defining valid bounds. The neighbors of a latent configuration z are the latent configurations that can be obtained by changing the value of one of the n latent variables in z.

Bound optimization: Optimization of a bound $b \in \mathcal{F}$ can be done in closed form by setting μ_j to be the mean of all examples assigned to cluster j:

$$\mu_j = \frac{\sum_{i \in I_j} x_i}{|I_j|}, \quad I_j = \{1 \leq i \leq n \mid z_i = j\}. \tag{14}$$

4.2 Latent Structural SVM

A Latent Structural SVM (LS-SVM) (Yu and Joachims, 2009) defines a structured output classifier with latent variables. It extends the Structural SVM (Joachims et al., 2009) by introducing latent variables.

Let $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ denote a set of labeled examples with $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. We assume that each example x_i has an associated latent value $z_i \in \mathcal{Z}$. Let $\phi(x, y, z) : \mathcal{X} \times \mathcal{Y} \times \mathcal{Z} \to \mathbb{R}^d$ denote a *feature map*. A vector $w \in \mathbb{R}^d$ defines a classifier $\hat{y} : \mathcal{X} \to \mathcal{Y}$,

$$\hat{y}(x) = \operatorname*{argmax}_y (\max_z w \cdot \phi(x, y, z)). \tag{15}$$

The LS-SVM training objective is defined as follows,

$$F(w) = \frac{\lambda}{2}||w||^2 + \frac{1}{n}\sum_{i=1}^{n}\Big(\max_{y,z}(w \cdot \phi(x_i, y, z) + \Delta(y, y_i)) - \max_z w \cdot \phi(x_i, y_i, z)\Big), \tag{16}$$

where λ is a hyper-parameter that controls regularization and $\Delta(y, y_i)$ is a non-negative loss function that penalizes the prediction y when the ground truth label is y_i.

Bound construction: As in the case of k-means, a convex upper bound on the LS-SVM objective can be obtained by imputing latent variables. Specifically, for each example x_i, we fix $z_i \in \mathcal{Z}$, and replace the maximization in the last term of the objective with a linear function $w \cdot \phi(x_i, y_i, z_i)$. This forms a family of convex piecewise quadratic bounds,

$$\mathcal{F} = \left\{\frac{\lambda}{2}||w||^2 + \frac{1}{n}\sum_{i=1}^{n}\max_{y,z}(w \cdot \phi(x_i, y, z) + \Delta(y, y_i)) - w \cdot \phi(x_i, y_i, z_i) \,\middle|\, \forall i, z_i \in \mathcal{Z}\right\}. \tag{17}$$

The CCCP algorithm for LS-SVM selects the bound b_t defined by $z_i^t = \operatorname{argmax}_{z_i} w_{t-1} \cdot \phi(x_i, y_i, z_i)$. This particular choice is a special case of G-MM when $g(b, w) = -b(w)$.

To generate random bounds from $\mathcal{B}_t$ we use the same approach as in the case of k-means clustering. We perform a random walk in a graph where the nodes are latent configurations leading to valid bounds, and the edges connect latent configurations that differ in a single latent variable.

Bound optimization: Optimization of a bound $b \in \mathcal{F}$ corresponds to a convex quadratic program and can be solved using different techniques, including gradient based methods (*e.g.* SGD) and the cutting-plane method (Joachims et al., 2009). We use the cutting-plane method in our experiments.

4.2.1 Bias Function for Multi-fold MIL

The multi-fold MIL algorithm (Cinbis et al., 2016) was introduced for training latent SVMs for weakly supervised object localization, to deal with stickiness issues in training with CCCP. It modifies how latent variables are updated during training. Cinbis et al. (2016) divide the training set into K folds, and updates the latent variables in each fold using a model trained on the other $K-1$ folds. This algorithm does not have a formal convergence guarantee. By defining a suitable bias function, we can derive a G-MM algorithm that mimics the behavior of multi-fold MIL, and yet, is convergent.

Consider training an LS-SVM. Let $S = (1, \ldots, n)$ be an ordered sequence of training sample indices. Also, let $z_i \in \mathcal{Z}$ denote the latent variable associated to training example (x_i, y_i). Let I denote a subsequence of S. Also, let z_I^t denote the fixed latent variable values of training examples in I in iteration t, and let $w(I, z_I^t)$ be the model trained on $\{(x_i, y_i)|i \in I\}$ with latent variables fixed to z_I^t in the last maximization of (16).

We assume access to a loss function $\ell(w, x, y, z)$. For example, for the binary latent SVM where $y \in \{-1, 1\}$, ℓ is the hinge loss: $\ell(w, x, y, z) = \max\{0, 1 - y\, w \cdot \phi(x, z)\}$.

We start by considering the Leave-One-Out (LOO) setting, *i.e.* $K = n$, and call the algorithm of Cinbis et al. (2016) LOO-MIL in this case. The update rule of LOO-MIL in iteration t is to set

$$z_i^t = \operatorname*{argmin}_{z \in \mathcal{Z}} \ \ell\left(w(S \backslash i, z_{S \backslash i}^{t-1}), x_i, y_i, z\right), \quad \forall i \in S. \tag{18}$$

After updating the latent values for all training examples, the model w is retrained by optimizing the resulting bound.

Now let us derive the bias function for a G-MM algorithm that mimics the behavior of LOO-MIL. Recall from Equation 17 that each bound $b \in \mathcal{B}_t$ is associated with a joint latent configuration $z(b) = (z_1, \ldots, z_n)$. We use the following bias function:

$$g(b, w) = -\sum_{i \in S} \ell\left(w(S \backslash i, z_{S \backslash i}^{t-1}), x_i, y_i, z_i\right). \tag{19}$$

Note that picking a bound according to (19) is equivalent to the LOO-MIL update rule of (18) except that in (19) only *valid* bounds are considered; that is bounds that make at least η-progress.

For the general multi-fold case (*i.e.* $K < n$), the bias function can be derived similarly.

5. Experiments

We evaluate G-MM and MM algorithms on k-means clustering and LS-SVM training on various datasets. Recall from (4) that the progress coefficient η defines the set of valid bounds $\mathcal{B}_t$ in each step. CCCP and standard k-means bounds correspond to setting $\eta = 1$, thus taking maximally large steps towards a local minimum of the true objective.

5.1 *k*-means Clustering

We conduct experiments on four different clustering datasets: Norm-25 (Arthur and Vassilvitskii, 2007), D31 (Veenman et al., 2002), Cloud (Arthur and Vassilvitskii, 2007), and GMM-200. Norm-25, D31, and GMM-200 are synthetic and Cloud is from real data. See the references for details about the datasets. GMM-200 was created by us. It is a Gaussian mixture model on 2-D data with 200 mixture components. Each component is a Gaussian distribution with $\sigma = 1.0$ and μ on a square of size 70×70. The means are placed at least 2.5σ apart from each other. The dataset contains 50 samples per mixture component.

We compare results from three different initializations: **forgy** selects k training examples uniformly at random without replacement to define initial cluster centers, **random partition** assigns training samples to cluster centers randomly, and ***k*-means++** uses the algorithm in (Arthur and Vassilvitskii, 2007). In each experiment we run the algorithm 50 times and report the mean, standard deviation, and the best objective value (Equation 12). Table 1 shows the results using k-means (hard-EM) and G-MM. We note that the variance of the solutions found by G-MM is typically smaller than k-means. Moreover, the best and the average solutions found by G-MM are always better than (or the same as) those found

dataset	k	opt. method	forgy		random partition		*k*-means++	
			avg ± std	best	avg ± std	best	avg ± std	best
Norm-25	25	hard-EM	$1.9e5 \pm 2e5$	$7.0e4$	$5.8e5 \pm 3e5$	$2.2e5$	$5.3e3 \pm 9e3$	1.5
		G-MM	$9.7e3 \pm 1e4$	1.5	$2.0e4 \pm 0$	$2.0e4$	$4.5e3 \pm 8e3$	1.5
D31	31	hard-EM	1.69 ± 0.03	1.21	52.61 ± 47.06	4.00	1.55 ± 0.17	1.10
		G-MM	1.43 ± 0.15	1.10	1.21 ± 0.05	1.10	1.45 ± 0.14	1.10
Cloud	50	hard-EM	1929 ± 429	1293	44453 ± 88341	3026	1237 ± 92	1117
		G-MM	1465 ± 43	1246	1470 ± 8	1444	1162 ± 95	1067
GMM-200	200	hard-EM	2.25 ± 0.10	2.07	11.20 ± 0.63	9.77	2.12 ± 0.07	1.99
		G-MM	2.04 ± 0.09	1.90	1.85 ± 0.02	1.80	1.98 ± 0.06	1.89

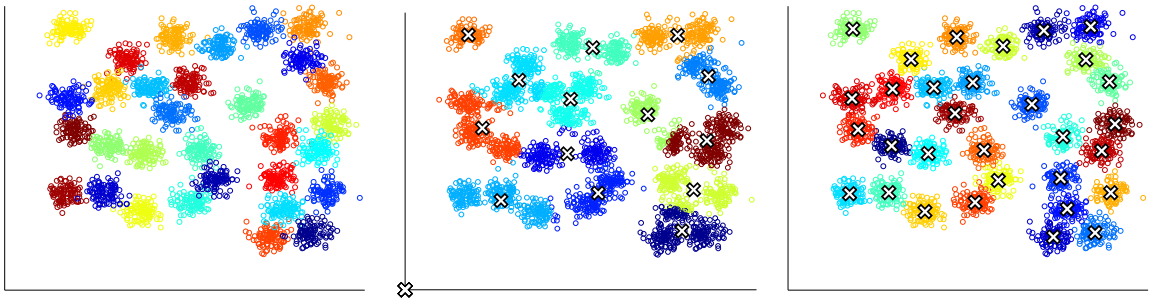
Table 1: Comparison of G-MM and k-means (hard-EM) on multiple clustering datasets. Three different initialization methods were compared; **forgy** initializes cluster centers to random examples, **random partition** assigns each data point to a random cluster center, and ***k*-means++** implements the algorithm from (Arthur and Vassilvitskii, 2007). The mean, standard deviation, and best objective values out of 50 random trials are reported. Both k-means and G-MM use the exact same initialization in each trial. G-MM consistently converges to better solutions.

by k-means. This trend generalizes over different initialization schemes as well as different datasets.

Although *random partition* seems to be a very bad initialization for k-means on all datasets, G-MM recovers from it. In fact, on D31 and GMM-200 datasets, G-MM initialized by *random partition* performs better than when it is initialized by other methods (including k-means++). Also, the variance of the best solutions (across different initialization methods) in G-MM is smaller than that of k-means. These suggest that the G-MM optimization is less sticky to initialization than k-means.

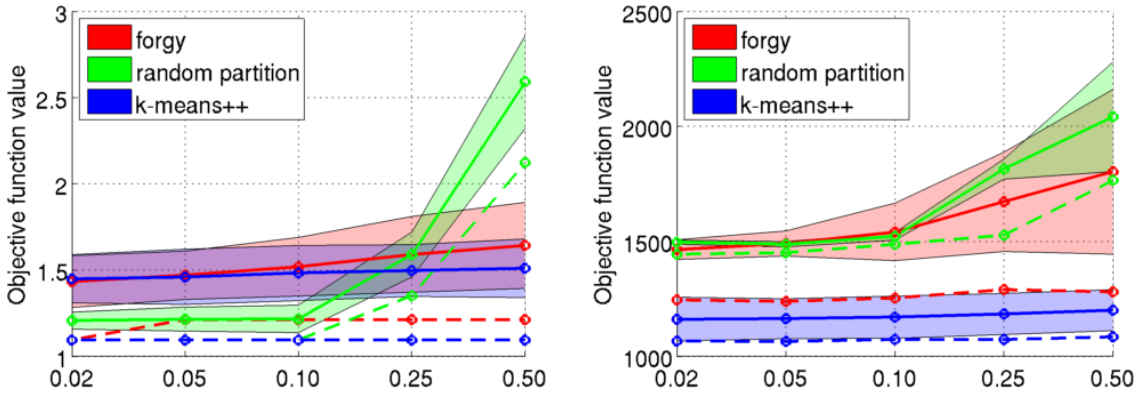
Figure 2 visualizes the result of k-means and G-MM (with random bounds) on the D-31 dataset (Veenman et al., 2002), from the same initialization. G-MM finds a near perfect solution, while in k-means many clusters get merged incorrectly or die off. Dead clusters are those which do not get any points assigned to them. The update rule of Equation (14) collapses the dead clusters on to the origin.

Figure 3 shows the effect of the progress coefficient on the quality of the solution found by G-MM. Different colors correspond to different initialization schemes. The solid line indicates the average objective over 50 iterations, the shaded area covers one standard deviation from the average, and the dashed line indicates the best solution over the 50 trials. Smaller progress coefficients allow for more extensive exploration, and hence, smaller variance in the quality of the solutions. On the other hand, when the progress coefficient is large G-MM is more sensitive to initialization (*i.e.* is more sticky) and, thus, the quality of the solutions over multiple runs is more diverse. However, despite the greater diversity, the best solution is worse when the progress coefficient is large. G-MM reduces to k-means if we set the progress coefficient to 1 (*i.e.* the largest possible value).



(a) ground truth (b) k-means (c) generalized-MM

Figure 2: Visualization of clustering solutions on the D31 dataset (Veenman et al., 2002) from identical initializations. Random partition initialization scheme is used. (a) color-coded ground-truth clusters. (b) solution of k-means. (c) solution of G-MM. The white crosses indicate location of the cluster centers. Color codes match up to a permutation.



(a) D31 (b) Cloud

Figure 3: Effect of the progress coefficient η (x-axis) on the quality of the solutions found by G-MM (y-axis) on two clustering datasets. The quality is measured by the objective function in (12). Lower values are better. The average (solid line), the best (dashed line), and the variance (shaded area) over 50 trials are shown in the plots and different initializations are color coded.

5.2 Latent Structural SVM for Image Classification and Object Detection

We consider the problem of training an LS-SVM classifier on the mammals dataset (Heitz et al., 2009). The dataset contains images of six mammal categories with image-level annotation. Locations of the objects in these images are not provided, and therefore, treated as latent variables in the model. Specifically, let x be an image and y be a class label ($y \in \{1, \ldots, 6\}$ in this case), and let z be the latent location of the object in the image. We define $\phi(x, y, z)$ to be a feature function with 6 blocks; one block for each category. It extracts features from location z of image x and places them in the y-th block of the output

Opt. Method	center		top-left		random	
	objective	test error	objective	test error	objective	test error
CCCP	1.21 ± 0.03	22.9 ± 9.7	1.35 ± 0.03	42.5 ± 4.6	1.47 ± 0.03	31.8 ± 2.6
G-MM random	0.79 ± 0.03	17.5 ± 3.9	0.91 ± 0.02	31.4 ± 10.1	0.85 ± 0.03	19.6 ± 9.2
G-MM biased	0.64 ± 0.02	16.8 ± 3.2	0.70 ± 0.02	18.9 ± 5.0	0.65 ± 0.02	14.6 ± 5.4

Table 2: LS-SVM results on the mammals dataset (Heitz et al., 2009). We report the mean and standard deviation of the training objective (Equation 16) and test error over five folds. Three strategies for initializing latent object locations are tried: image center, top-left corner, and random location. "G-MM random" uses random bounds, and "G-MM bias" uses a bias function inspired by multi-fold MIL (Cinbis et al., 2016). Both variants consistently and significantly outperform the CCCP baseline.

and fills the rest with zero. We use the following multi-class classification rule:

$$y(x) = \underset{y,z}{\operatorname{argmax}}\, w \cdot \phi(x, y, z),\ w = (w_1, \ldots, w_6). \tag{20}$$

In this experiment we use a setup similar to that in (Kumar et al., 2012): we use Histogram of Oriented Gradients (HOG) for the image feature ϕ, and the 0-1 classification loss for Δ. We set $\lambda = 0.4$ in Equation 16. We report 5-fold cross-validation performance. Three initialization strategies are considered for the latent object locations: *image center*, *top-left corner*, and *random locations*. The first is a reasonable initialization since most objects are at the center in this dataset; the second initialization strategy is somewhat adversarial.

We try a stochastic as well as a deterministic bound construction method. For the stochastic method, in each iteration t we uniformly sample a subset of examples S_t from the training set, and update their latent variables using $z_i^t = \operatorname{argmax}_{z_i} w_{t-1} \cdot \phi(x_i, y_i, z_i)$. Other latent variables are kept the same as the previous iteration. We increase the size of S_t across iterations.

For the deterministic method, we use the bias function that we described in Section 4.2.1. This is inspired by the multi-fold MIL idea (Cinbis et al., 2016) and is shown to reduce stickiness to initialization, especially in high dimensions. We set the number of folds to $K = 10$ in our experiments.

Table 2 shows results on the mammals dataset. Both variants of G-MM consistently outperform CCCP in terms of training objective and test error. We observed that CCCP rarely updates the latent locations, under all initializations. On the other hand, both variants of G-MM significantly alter the latent locations, thereby avoiding the local minima close to the initialization. Figure 4 visualizes this for *top-left* initialization. Since objects rarely occur at the top-left corner in the mammals dataset, a good model is expected to significantly update the latent locations. Averaged over five cross-validation folds, about 90% of the latent variables were updated in G-MM after training whereas this measure was 2.4% for CCCP. This is consistent with the better training objectives and test errors of G-MM.

We provide additional experimental results on the mammals dataset. Figure 5 shows example training images and the final imputed latent object locations by three algorithms: CCCP (red), G-MM random (blue), and G-MM biased (green). The initialization is *top-left*.

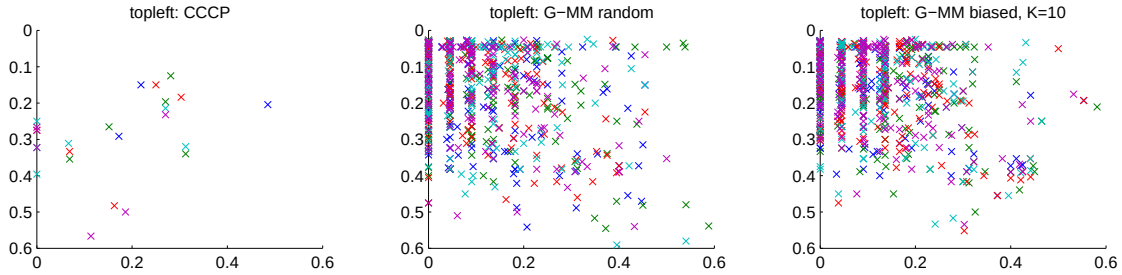


Figure 4: Latent location changes after learning, in relative image coordinates, for all five cross-validation folds, for the *top-left* initialization on the mammals dataset. Left to right: CCCP, "G-MM random", "G-MM biased" (K=10). Each cross represents a training image; cross-validation folds are color coded differently. Averaged over five folds, CCCP only alters 2.4% of all latent locations, leading to very bad performance. "G-MM random" and "G-MM biased" alter 86.2% and 93.6% on average, respectively, and perform much better.

In most cases CCCP fails to update the latent locations given by initialization. The two G-MM variants, however, are able to update them significantly and often localize the objects in training images correctly. This is achieved *only* with image-level object category annotations, and with a very bad (even adversarial) initialization.

5.3 Latent Structural SVM for Scene Recognition

We implement the reconfigurable model of Parizi et al. (2012) to do scene classification on MIT-Indoor dataset (Quattoni and Torralba, 2009), which has images from 67 indoor scene categories. We segment each image into a 10×10 regular grid and treat the grid cells as image regions. We train a model with 200 shared parts. All parts can be used to describe the data in an image region. We use the pre-trained hybrid ConvNet from Zhou et al. (2014) to extract features from image regions. We record from the 4096 neurons at the penultimate layer of the network and use PCA to reduce the dimensionality of these features to 240.

The reconfigurable model is an instance of LS-SVM models. The latent variables are the assignments of parts to image regions and the output structure is the multi-valued category label predictions. LS-SVMs are known to be sensitive to initialization (*a.k.a.* the stickiness issue). Parizi et al. (2012) cope with this issue by training a generative version of the model first and using it to initialize the discriminative model. Generative models are typically less sticky but perform worse in practice. To validate the hypothesis regarding stickiness of LS-SVMs we run training using several initialization strategies.

Initializing training entails the assignment of parts to image regions *i.e.* setting z_i's in Equation 17 to define the first bound. To this end we first discover 200 parts that capture discriminative features in the training data. We then run graph cut on each training image to obtain part assignments to image regions. Each cell in the 10×10 image grid is a node in the graph. Two nodes in the graph are connected if their corresponding cells in the image grid are next to each other. Unary terms in the graph cut are the dot product scores between the feature vector extracted from an image region and a part filter plus the corresponding region-to-part assignment score. Pairwise terms in the graph cut implement

Figure 5: Example training images from the mammals dataset, shown with final imputed latent object locations by three algorithms: CCCP (red), G-MM random (blue), G-MM biased (green). Initialization: *top-left*.

Opt. Method	Random		$\lambda = 0.00$		$\lambda = 0.25$		$\lambda = 0.50$		$\lambda = 1.00$	
	Acc.% $\pm$ std	O.F.	Acc. %	O.F.	Acc. %	O.F.	Acc. %	O.F.	Acc. %	O.F.
CCCP	41.94 ± 1.1	15.20	40.88	14.81	43.99	14.77	45.60	14.72	46.62	14.70
G-MM random	47.51 ± 0.7	14.89	43.38	14.71	44.41	14.70	47.12	14.66	49.88	14.58
G-MM biased	49.34 ± 0.9	14.55	44.83	14.63	48.07	14.51	53.68	14.33	56.03	14.32

Table 3: Performance of LS-SVM trained with CCCP and G-MM on MIT-Indoor dataset. We report classification accuracy (Acc.%) and the training objective value (O.F.). Columns correspond to different initialization schemes. "Random" assigns random parts to regions. λ controls the coherency of the initial part assignments: $\lambda = 1$ ($\lambda = 0$) corresponds to the most (the least) coherent case. "G-MM random" uses random bounds and "G-MM biased" uses the bias function of Equation 21. $\eta = 0.1$ in all the experiments. Coherent initializations lead to better models in general, but, they require discovering good initial parts. "G-MM" outperforms CCCP, especially with random initialization. "G-MM biased" performs the best.

a *Potts* model that encourages coherent labelings. Specifically, the penalty of labeling two neighboring nodes differently is λ and it is zero otherwise. λ controls the coherency of the initial assignments. We experiment using $\lambda \in \{0, 0.25, 0.5, 1\}$. We also experiment with random initialization, which corresponds to assigning z_i's randomly. This is the simplest form of initialization and does not require discovering initial part filters.

We do G-MM optimization using both random and biased bounds. For the latter we use a bias function $g(b, w)$ that measures coherence of the labeling from which the bound was constructed. Recall from Equation 17 that each bound in $b \in B_t$ corresponds to a labeling of the image regions. We denote the labeling corresponding to the bound b by $z(b) = (z_1, \ldots, z_n)$ where $z_i = (z_{i,1}, \ldots, z_{i,100})$ specifies part assignments for all the 100 regions in the i-th image. Also, let $E(z_i)$ denote a function that measures coherence of the labeling z_i. In fact, $E(z_i)$ is the Potts energy function on a graph whose nodes are $z_{i,1}, \ldots, z_{i,100}$. The graph respects a 4-connected neighborhood system (recall that $z_{i,r}$ corresponds to the r-th cell in the 10×10 grid defined on the i-th image). If two neighboring nodes $z_{i,r}$ and $z_{i,s}$ get different labels the energy $E(z_i)$ increases by 1. For biased bounds we use the following bias function which favors bounds that correspond to more coherent labelings:

$$g(b, w) = -\sum_{i=1}^{n} E(z_i), \qquad z(b) = (z_1, \ldots, z_n). \tag{21}$$

Table 3 compares performance of models trained using CCCP and G-MM with random and biased bounds. For G-MM with random bounds we repeat the experiment five times and report the average over these five trials. Also, for random initialization, we do five trials using different random seeds and report the mean and standard deviation of the results. G-MM does better than CCCP under *all* initializations. It also converges to a solution with lower training objective value than CCCP. Our results show that picking bounds uniformly at random from the set of valid bounds is slightly (but consistently) better than committing to the CCCP bound. We get a remarkable boost in performance when we use a reasonable prior over bounds (*i.e.* the bias function of Equation 21). With $\lambda = 1$, CCCP attains

experiment	setup	MM	G-MM	
			random	biased
scene recognition	$\lambda = 0.0$	145	107	87
	$\lambda = 1.0$	65	69	138
data clustering (GMM-200)	forgy	35.76 ± 7.8	91.52 ± 4.4	
	rand. part.	114.98 ± 12.9	241.89 ± 2.1	
	k-means++	32.92 ± 5.8	80.78 ± 2.9	
data clustering (Cloud)	forgy	37.18 ± 12.1	87.68 ± 15.4	
	rand. part.	65.14 ± 18.7	138.64 ± 5.9	
	k-means++	21.3 ± 4.1	44.12 ± 10.7	

Table 4: Comparison of the number of iterations it takes for MM and G-MM to converge in the scene recognition and the data clustering experiment with different initializations and/or datasets. The numbers reported for the clustering experiment are the average and standard deviation over 50 trials.

accuracy of 46.6%, whereas G-MM attains 49.9%, and 56.0% accuracy with random and biased initialization respectively. Moreover, G-MM is less sensitive to initialization.

5.4 Running Time

G-MM bounds make a fraction of the progress made in each bound construction step compared to the MM bound. Therefore, we would expect G-MM to take more steps before it converges. We report the number of iterations that MM and G-MM take to converge in Table 4. The results for G-MM depend on the value of the progress coefficient η which is set to match the experiments in the paper; $\eta = 0.02$ for the clustering experiment (Section 5.1) and $\eta = 0.10$ for the scene recognition experiment (Section 5.3).

The overhead of the bound construction step depends on the application. For example, in the scene recognition experiment, optimizing the bounds takes orders of magnitude more than sampling them (a couple of hours vs. a few seconds). In the clustering experiment, however, the optimization step is solved in closed form whereas sampling a bound involves performing a random walk on a large graph which can take a couple of minutes to run.

6. Conclusion

We have introduced Generalized Majorization-Minimization (G-MM), a generic iterative bound optimization framework that generalizes upon Majorization-Minimization (MM). Our key observation is that MM enforces an overly-restrictive touching constraint in its bound construction mechanism, which is inflexible and can lead to sensitivity to initialization. By adopting a different measure of progress, in G-MM, this constraint is relaxed to allow for more freedom in bound construction. Specifically, we propose deterministic and stochastic ways of selecting bounds from a set of valid ones. This generalized bound construction process tends to be less sensitive to initialization, and enjoys the ability to directly incorporate rich application-specific priors and constraints, without modifications to the objective function. In experiments with several latent variable models, G-MM algorithms are shown to significantly outperform their MM counterparts.

Future work includes applying G-MM to a wider range of problems and theoretical analysis, such as convergence rate. We also note that, although G-MM is more conservative than MM in moving towards nearby local minima, it still requires *making progress* in every step. Another interesting research direction is to enable G-MM to occasionally pick bounds that do not make progress with respect to the solution of the previous bound, thereby making it possible to get out of local minima, while still maintaining the convergence guarantees of the method.

Appendix A. Proof of Theorem 3

We have proved two theorems in the main paper. In Theorem 1, we showed that $\lim_{t\to\infty} d_t = 0$, where $d_t = b_t(w_t) - F(w_t)$ is the gap between the G-MM bound and the objective function at iteration t at the minimizer of the bound. In Theorem 2, we showed that G-MM converges to a solution, or $\exists w^\dagger$ *s.t.* $\lim_{t\to\infty} w_t = w^\dagger$.

In this appendix we show that under under mild conditions, our G-MM framework converges to a local extremum or a saddle point of the objective function. Theorem 3 roughly states that, if F is continuously differentiable, and the family of bounds are smooth and have bounded curvature, then G-MM converges to a stationary point; that is, $\nabla F(w^\dagger) = 0$.

Proof. We briefly overview the sketch of the proof first before explaining the steps in detail. We prove the theorem by contradiction by showing that if $\nabla F(w^\dagger) \neq 0$ then we can construct a lower bound on F (stated in (29)) that, at some point w, goes above an upper bound of b_t (stated in (33)). This implies that $\exists w$ *s.t.* $F(w) > b_t(w)$ which contradicts with the assumption that b_t is an upper bound of F. The proof has three steps: 1) constructing the lower bound on F, 2) constructing the upper bound on b_t, and 3) finding a point w *s.t.* $F(w) > b_t(w)$. Details are presented below.

Step 1: Assume $||\nabla F(w^\dagger)||_2 \neq 0$. This can be stated as

$$\exists A > 0 \text{ s.t. } ||\nabla F(w^\dagger)||_2 = 2A. \tag{22}$$

Consider value of F along the gradient direction at $w^\dagger$. We denote this by $g(z)$ where $z \in \mathbb{R}$ and define it as

$$g(z) = F(w^\dagger + zu), \quad u = \frac{\nabla F(w^\dagger)}{||\nabla F(w^\dagger)||_2}. \tag{23}$$

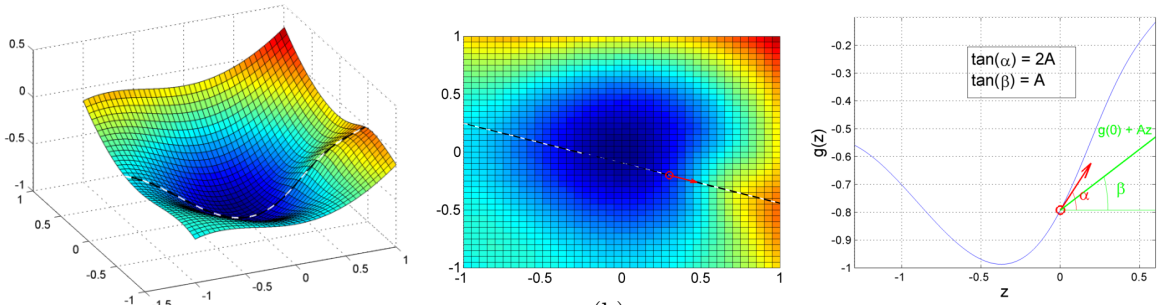
In what follows, we only consider the case where $z \geq 0$. We can write the following equalities:

$$g'(z) = \frac{\partial g}{\partial z}(z) = \nabla F(w^\dagger + zu) \cdot u \tag{24}$$

$$g'(0) = ||\nabla F(w^\dagger)||_2 = 2A \tag{25}$$

$$g(0) = F(w^\dagger). \tag{26}$$

In Figure 6a we show a non-convex function F in $\mathbb{R}^2$. The gradient direction at a particular point p is also shown in the figure. Figure 6b shows the same function from the top view, together with the gradient direction at p. Figure 6c shows how the value of F



(a) (b) (c)

Figure 6: An example of a non-convex function F on $\mathbb{R}^d$. (a) shows the plot of F; the value of the function on the line that touches F at the point $p = (0.3, -0.2)$ is marked by the black and white line. (b) shows F from the top view; it also marks the point p with the red circle and shows the direction of the gradient vector at p. The length of the vector is set arbitrarily for visualization purposes. (c) shows the value of F along the tangent line as a one dimensional function (see Equation 23); it also shows the gradient direction at $z = 0$ (in red) and the linear function of Equation 29 that lower bounds F (in green).

changes along the tangent line as a function of the step size z along the gradient direction u (see $g(z)$ in Equation 23).

Since F is differentiable, we can use the mean value theorem to rewrite g for $z \geq 0$ as

$$\forall z \geq 0, \exists k \in [0, z] \ s.t. \ g(z) = g(0) + g'(k)z. \tag{27}$$

Further, since we assume F to be continuously differentiable, so is g. Continuity of g' implies that in a small neighborhood around 0, g' is larger than A:

$$\exists \delta > 0 \ s.t. \ \forall z \in [0, \delta], \ g'(z) > A. \tag{28}$$

We can use this inequality in (27) and get a lower bound on g in the neighborhood of 0, or equivalently, a lower bound on F in the vicinity of $w^\dagger$ and in the direction of u:

$$\begin{aligned} & \exists \delta > 0, \forall z \in [0, \delta], \ g(z) > g(0) + Az \\ \Rightarrow \ & \exists \delta > 0, \forall z \in [0, \delta], \ F(w^\dagger + zu) > F(w^\dagger) + Az. \end{aligned} \tag{29}$$

This lower bound is shown in Figure 6c in green.

Step 2: To get an upper bound on b_t, we note that $\nabla b_t(w_t) = 0$, since $w_t = \text{argmin}_w \, b_t(w)$ is the minimizer of b_t and b_t is differentiable. From this and the assumption that the curvature of all bounds in all directions is finite (*i.e.* $\forall b \in \mathcal{F}, \nabla^2 b(w) \preceq MI$) we can upper bound b_t with a quadratic function with curvature M whose minimum occurs at $w = w_t$ and has value $b_t(w_t)$, as follows:

$$b_t(w) \leq (w - w_t)^T MI(w - w_t) + b_t(w_t), \forall w. \tag{30}$$

Similar to step 1, we only focus on the points w that lie on the line $w^\dagger + zu$ and get the following bound on b_t:

$$b_t(w^\dagger + zu) \leq M||w^\dagger - w_t + zu||^2 + b_t(w_t) \tag{31}$$
$$\leq 2M||w^\dagger - w_t||^2 + 2Mz^2 + b_t(w_t), \forall z \in \mathbb{R}. \tag{32}$$

Finally, from Theorem 2 we have that $\forall \epsilon_1 > 0$, $\exists T_1$ *s.t.* $\forall t > T_1$, $||w^\dagger - w_t||_2 < \epsilon_1$. We can use this in (32) to get the following strict upper bound:

$$b_t(w^\dagger + zu) < 2M\epsilon_1^2 + 2Mz^2 + b_t(w_t), \forall t > T_1, \forall z \in \mathbb{R}. \tag{33}$$

Step 3: We show that $\exists z \in [0, \delta]$ for which the right hand side of (29) is larger than the right hand side of (33), which leads to the contradiction that for $w = w^\dagger + zu$, $F(w) > b_t(w)$. In other words, we need to find $z \in [0, \delta]$ such that $F(w^\dagger) + Az > 2M\epsilon_1^2 + 2Mz^2 + b_t(w_t)$, or

$$2Mz^2 - Az + 2M\epsilon_1^2 + b_t(w_t) - F(w^\dagger) < 0. \tag{34}$$

Lemma 1. $\forall \epsilon_2 > 0, \exists T_2$ s.t. $\forall t > T_2, b_t(w_t) - F(w^\dagger) < \epsilon_2$.

Proof. To prove the lemma we first show that

$$\lim_{t \to \infty} |b_t(w_t) - F(w^\dagger)| = 0, \tag{35}$$

and since $b_1(w_1), b_2(w_2), \ldots$ is a non-increasing sequence, $\forall w$, $b_t(w_t) \geq F(w^\dagger)$, thus we can drop the absolute value.

To show (35), we can write

$$|b_t(w_t) - F(w^\dagger)| \leq |b_t(w_t) - F(w_t)| + |F(w_t) - F(w^\dagger)|.$$

The first term on the right hand side can get arbitrarily close to zero due to Theorem 1. The second term, also, can get arbitrarily close to zero due to Theorem 2 and that F is Lipschitz continuous (because it is continuously differentiable). □

Applying lemma 1 to inequality (34), now we only need to show that

$$\exists z \in [0, \delta],\ 2Mz^2 - Az + 2M\epsilon_1^2 + \epsilon_2 < 0. \tag{36}$$

Note that ϵ_1 and ϵ_2 can be chosen arbitrarily. In particular, we can set them so that the discriminant of the quadratic function in (36) is positive:

$$\Delta = A^2 - 4\,(2M)\left(2M\epsilon_1^2 + \epsilon_2\right) > 0 \Rightarrow 2M\epsilon_1^2 + \epsilon_2 < \frac{A^2}{8M}. \tag{37}$$

This guarantees that the quadratic function in (36) has two distinct roots. If $\epsilon_1 = \epsilon_2 = 0, 2M\epsilon_1^2 + \epsilon_2 = 0$, the two roots are $z_1 = 0$ and $z_2 = \frac{A}{2M} > 0$. For $\epsilon_1, \epsilon_2 > 0$, only the constant $2M\epsilon_1^2 + \epsilon_2$ is affected which shifts the quadratic function up. We can now control ϵ_1 and ϵ_2 to drive the constant arbitrarily close to 0, causing $z_1 > 0$ to get arbitrarily close to zero, and thus fall within the interval $[0, \delta]$. Therefore (34) can be satisfied.

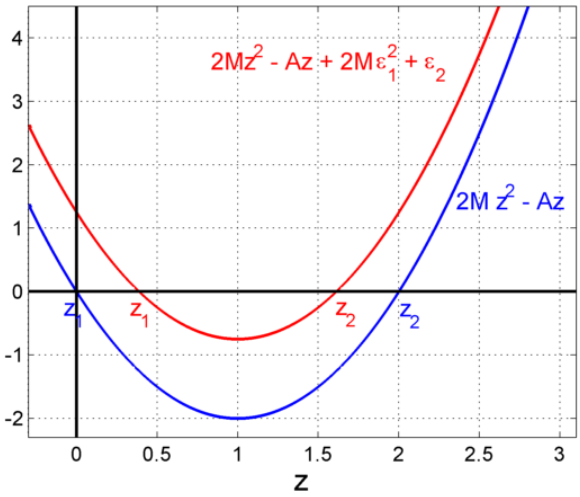


Figure 7: The quadratic function in the left hand side of (36). When $\epsilon_1 = \epsilon_2 = 0$, the quadratic function has two roots, $z_1 = 0$ and $z_2 = \frac{A}{2M} > 0$ (shown in blue). When $\epsilon_1, \epsilon_2 > 0$ but properly chosen, the quadratic function is shifted up but still has two distinct roots (shown in red).

To summarize, if we assume $\|\nabla F(w^\dagger)\|_2 = 2A > 0$, then we have a lower bound on F along the gradient direction u at $w^\dagger$ (step 1): $\exists \delta > 0, \forall z \in [0, \delta], F(w^\dagger + zu) > F(w^\dagger) + Az$. Along the same direction, b_t can be upper bounded for sufficiently large t (step 2): $\forall \epsilon_1 > 0, \exists T_1 \ s.t. \ \forall t > T_1, b_t(w^\dagger + zu) < 2M\epsilon_1^2 + 2Mz^2 + b_t(w_t)$. Finally, we can find $z_1 \in [0, \delta]$ such that $F(w^\dagger) + Az_1 > 2M\epsilon_1^2 + 2Mz_1^2 + b_t(w_t)$ (step 3), leading to $F(w^\dagger + z_1 u) > b_t(w^\dagger + z_1 u)$. However, this violates the assumption that b_t is an upper bound of F, therefore $||\nabla F(w^\dagger)||_2 = 0$. □

References

David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *Symposium on Discrete Algorithms (SODA)*, 2007.

Hossein Azizpour, Mostafa Arefiyan, Sobhan Naderi, and Stefan Carlsson. Spotlight the negatives: A generalized discriminative latent model. In *British Machine Vision Conference (BMVC)*, 2015.

Ramazan Gokberk Cinbis, Jakob Verbeek, and Cordelia Schmid. Weakly Supervised Object Localization with Multi-fold Multiple Instance Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, January 2016.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

Pedro Felzenszwalb, Ross Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2010.

Kuzman Ganchev, Joao Graça, Jennifer Gillenwater, and Ben Taskar. Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research*, 11:2001–2049, 2010.

Asela Gunawardana and William Byrne. Convergence theorems for generalized alternating minimization procedures. *The Journal of Machine Learning Research*, 6:2049–2073, 2005.

Geremy Heitz, Gal Elidan, Benjamin Packer, and Daphne Koller. Shape-based object localization for descriptive classification. *International journal of computer vision (IJCV)*, 2009.

David Hunter, Kenneth Lange, and Ilsoon Yang. Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, 2000.

Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 2009.

M Pawan Kumar, Ben Packer, and Daphne Koller. Modeling latent variable uncertainty for loss-based learning. In *International Conference on Machine Learning (ICML)*, 2012.

Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

James MacQueen. Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

Radford Neal and Geoffrey Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in Graphical Models*, 1998.

Sobhan Naderi Parizi, John Oberlin, and Pedro Felzenszwalb. Reconfigurable models for scene recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. Inducing features of random fields. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(4):380–393, 1997.

Wei Ping, Qiang Liu, and Alexander Ihler. Marginal structured SVM with hidden variables. In *International Conference on Machine Learning (ICML)*, 2014.

Hamed Pirsiavash and Deva Ramanan. Parsing videos of actions with segmental grammars. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

Mohammad Rastegari, Hannaneh Hajishirzi, and Ali Farhadi. Discriminative and consistent similarities in instance-level multiple instance learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Christian Ries, Fabian Richter, and Rainer Lienhart. Towards automatic bounding box annotations from weakly labeled images. *Multimedia Tools and Applications*, 2015.

Ruslan Salakhutdinov, Sam Roweis, and Zoubin Ghahramani. On the convergence of bound optimization algorithms. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence (UAI)*, pages 509–516. Morgan Kaufmann Publishers Inc., 2002.

Hyun Oh Song, Ross Girshick, Stefanie Jegelka, Julien Mairal, Ziad Harchaoi, and Trevor Darrell. On learning to localize objects with minimal supervision. In *International Conference on Machine Learning (ICML)*, 2014.

Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Pseudo-bound optimization for binary energies. In *European Conference on Computer Vision (ECCV)*. Springer, 2014.

Pham Dinh Tao. Convex analysis approach to dc programming: Theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

Cor J. Veenman, Marcel Reinders, and Eric Backer. A maximum variance cluster algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2002.

Eric P. Xing, Michael I. Jordan, Stuart J Russell, and Andrew Y. Ng. Distance metric learning with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 521–528. MIT Press, 2002.

Chun-Nam Yu. Transductive learning of structural svms via prior knowledge constraints. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1367–1376, 2012.

Chun-Nam John Yu and Thorsten Joachims. Learning structural SVMs with latent variables. In *International Conference on Machine Learning (ICML)*. ACM, 2009.

AL Yuille and A Rangarajan. The concave-convex procedure. *Neural computation*, 2003.

Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems (NIPS)*, 2014.